AMENDMENTS TO THE SPECIFICATION

Please replace the paragraph on page 20, lines 4-15 with the following paragraph:

FIG. 1 illustrates a system architecture for enabling application-to-application interaction in a distributed computer network. Specifically, the system architecture of FIG. 1 illustrates an inter- or intra-enterprise Internet-based electronic commerce architecture including process automation application 10, referred to as a commerce exchange (CX) server. CX server 10 operates as a type of clearinghouse, receiving operation requests posted by client components 34 and 38 and directing them to appropriate service components 26, 28 and 30 (via communication connections 25, 27, and 29, respectively) identified ("signed up") to CX server 10 as being available to perform those services. In this capacity, much of the processing performed by CX server 10 involves searching for service component by service operation and searching for client components by their identification numbers. CX server 10 also performs a variety of administrative functions including transaction tracking and audit functions and disaster recovery functions.

Please replace the paragraph on page 38, lines 6-18 with the following paragraph:

A transaction instance 52 is composed of a set of ordered operations 54. In the directed ~~acylic~~ acyclic graph, an operation is represented by a node in the graph. Every transaction has two specific nodes, or operations, called the head operation and the tail operation. Operations 61 and 63 are shown as the head and tail operations respectively. Operation flow within a transaction always proceeds from the head operation to the tail operation. There may be one or more operations between the head and tail operations but each operation is performed only once. Note that if the operation is a broadcast operation, it is still considered to be performed only once, even though the operation may be sent to many

service applications to be performed. There may be more than one possible path through the graph from the head operation to the tail operation, and one of those possible paths is executed at runtime. Thus, the path through the graph for a given transaction definition will not necessarily be the same for each transaction instance of that transaction definition because of differing run time conditions.

Please replace the paragraph on page 41, lines 10-15 with the following paragraph:

In an illustrated embodiment of the present invention, the transaction DAG data structure 82 has the structure of the document type definition (DTD) shown in Table 1 and illustrated in FIG. 6 and FIG 7. The INPUT entity 60, CONDITIONAL LOGIC entity 58 and OPERATION LINK(S) entity 56 of FIG. 5 are referred to as JOIN section 87, SPLIT section 86, and OPLINK section 85, respectively, in Table 1 and in FIG. 6 and FIG. 7.

Please replace the paragraph on page 52, lines 7-22 with the following paragraph:

A join may be comprised of a series of argument sections (ARG) 92 which specify the value to be given to one or more specific named variables in the operation request document of the operation. The ARG section 92 consists of a series of tags defining the properties of the variable(s) in the operation request document and either an expression or document mapping to give the variable its value. Each variable is given a name (NAME) and a representation type (VALTYPE) along with an optional tag indicating if this variable is mandatory or optional (OPTIONAL). If no OPTIONAL tag is provided, or the value of the OPTIONAL tag is "NO" (or "no"), the variable is considered mandatory. If the OPTIONAL tag is present and its value is either "YES" (or "yes"), the variable is considered optional. This is used to determine how to proceed if the named variable cannot be given a value. If an optional variable is not available, the join can still succeed, provided that any other mandatory

variables are available. If a mandatory variable is not available, the join is considered to have failed and the operation cannot be executed. Failure of the operation may, in turn, cause the entire transaction to fail. The variables may be given values based on the evaluation of some expression (EXPR) or from a mapped document, or specific section of a document.

Please replace the paragraph on page 59, lines 2-15 with the following paragraph:

The EXPR section 94 is used for split conditions, for determining values for arguments in a JOIN, and for defining BROADCAST advance criteria. An expression may be a simple value (VALUE) 95, an operation (OPERATOR) 96, a function (FUNCTION) 98, or the value of a specified variable (VAR) 97. When an expression is a simple value, whenever the expression is evaluated, it always returns the configured value. The VALUE parameter of the EXPR section could be used to initialize some variable in an operation to a configured value. An expression may be defined as the value of a named variable using the VAR tag. Whenever the expression is evaluated, the value of the named variable is returned, or an error is returned if the variable could not be found. Using the VAR tag in an expression also requires the following additional information: the name (VARNAME) of the variable; the ID of the operation that contains the variable (OPID); the document that contains the variable (OPDOCIOTYPE); the section of that document that contains the variable (OPDOCVARTYPE); and the representation type of the value (VALTYPE).